REMARKS

The applicant thanks the examiner for the telephone interview held on 11 June 2007 with the applicant's representative, David Feigenbaum, and his colleague, Frank Gerratana.

In the interview, Mr. Feigenbaum pointed out that each of the "regions" in claim 1 contains data that is "all either locked or not locked for writing at a given time," a feature that the cited portions of Hapner simply did not describe and would not have made obvious.

The examiner also speculated in the interview that the word regions in claim 1 could be construed broadly enough to include, for example, portions of a distributed database that are located in different cities and that Hapner's database could have had such database portions and therefore have met the language of claim 1.

Although the examiner confirmed that an amendment to define regions more specifically might be useful, the applicant now believes that no clarification of regions is needed and has instead amended claim 1 to recite that each of the regions is assigned "to a respective available processor, each of the regions being assignable to any of the processors." The cited portions of Hapner and Somani give no hint of assigning such regions to processors, let alone of being able to assign a region to any of the processors.

Additional comments of the applicant below are each preceded by related comments of the examiner (in small, bold type).

6. The disclosure is objected to because of the following informalities: The arrangement of the specification is not in the required order. Appropriate correction is required.

The applicant disagrees, but has amended the specification for convenience.

9. Claims 57-61 in view of the above cited MPEP section, are not statutory because claims they merely recite computing steps without producing any concrete and useful result and/or being limited to a practical application within the technological arts. Claim 57 lacks the necessary physical articles or objects to constitute a machine or a manufacture within the meaning of 35 USC 101. They are clearly not a series of steps or acts to be a process nor are they a combination of chemical compounds to be a composition of matter. As such, they fail to fall within a statutory category. They are, at best, functional descriptive material per so. Descriptive material can be characterized as either "functional descriptive material" or "nonfunctional descriptive material." Both types of "descriptive material" are nonstatutory when claimed as descriptive material per so, 33 F.3d at 1360, 31 USPQ2d at 1759. When functional descriptive material is recorded on some computer-readable medium, it becomes structurally and functionally interrelated to the medium and will be statutory in most cases since use of technology permits the function of the descriptive material to be realized. Compare In re Lowry, 32 F.3d 1579, 1583-84, 32 USPQ2d 1031,1035 (Fed. Cir. 1994). Merely claiming nonfunctional descriptive material, i.e., abstract ideas, stored on a computer-readable medium, in a computer, or on an electromagnetic carrier signal, does not make it statutory. See Diehr, 450 U.S. at 185-86, 209 USPQ at 8 (noting that the claims for an algorithm in Benson were unpatentable as abstract ideas because "[t]he sole practical application of the algorithm was in connection with the programming of a general purpose computer.")

The applicant disagrees, but has amended claim 57.

Claim 52 is the system claim recites similar limitations as to claim 1; therefore, claim 52 is rejected under the same reason as to claim 1.

The applicant disagrees. Certain features of claim 52 are different from claim 1, for example, the conflicting contention spaces. The applicant respectfully asks the examiner to address all of the features of claim 52 independently of his rejection of claim 1.

11. Claims 57-61 are rejected under 35 U.S.C. 103(a) as being unpatentable over Hapner et al. (US. Patent No. 5,727,203) in view of Zaiken et al. (US. Patent No 5,907,848).

Regarding on claim 57, Hapner teaches a software object configured to be executed on a machine, the object comprising:

A job to be executed (threads), the job requiring access (write to the database 159) (col. 9, lines 25-30)to a region of a database that stores data persistently (persistence data portion 164) (col. 9, lines 55-60), the job including instruction and pointers to data in the region of the database (col. 9, lines 55-60);

Hapner teaches "as corresponding to multiples threads competing for the resource of both the database cache 160 and the persistence database portion 164" (col. 9, lines 55-60). Hapner does not explicitly teach an index that identifies a contention space of jobs that have competing requirement to write into the region of the database, the index distinguish the contention space from other contention spaces of jobs that do not have competing requirements to write into the region of the database. On the other hand, Zaiken teaches an index that identifies a contention space of jobs that have competing requirement to write into the region of the database, the index distinguish the contention space from other contention spaces of jobs that do not have competing requirements to write into the region of the database "as corresponding to if the file name in the records 20 matches a filename in the selected template 28, the transaction monitor program then searches for any existing index files 30 having job identifier data equal to the job identifier data in the record 20" (col. 11, lines 39-43). This teaches the index identifying the job in the index files to distinguish by comparing the job identifier in the file. Therefore, it would have been obvious to one ordinary skill in the art at the time of l:he invention was made to modify Hapner system to include comparing the job identifier in the index files of Zaiken on order to distinguish the jobs by comparing the job identifier in order to process the requested job.

The applicant disagrees. Claim 57 recites the contention index, which identifies "a contention space of jobs, the index distinguishing the contention space from other contention spaces of jobs that do not have competing requirements to write into the region of the database." Zaiken does not have an index that distinguishes contention spaces, and describes no mechanism for associating database regions with categories of jobs.

Zaiken's index files are vehicles for indicators that identify only an individual database transaction, not a category of transactions associated with a particular region of the database. (Col. 4, lines 3-7; Col. 4, lines 33 – 34). Thus, no combination of Hapner and Zaiken would have disclosed or suggested "an index that identifies a contention space of jobs that have competing requirement to write into the region of the database, the index distinguishing the contention space from other contention spaces of jobs that do not have competing requirements to write into the region of the database."

Regarding on claim 68, Hapner teaches a method comprising:

Maintaining a database that stores data persistently (col. 7, lines 15-16) and provides a primary level of guarantee that data written in a request transaction is not lost once the transaction is committed (col. 10, lines 11-22),
Accepting tasks from the task source for concurrent execution by multiple processors, at least some of the tasks having conflicting requirements to write into the same region of the database (col. 9, lines 54-65), and

Hapner teaches, "a mutex is created to corresponding to a piece of code, a portion data, some state, etc... when a thread has locked a mutex, it is said to "own" the locked mutex. In order for other threads to own the mutex, the first thread (i.e. the thread that locked the rnutex) must unlock it. Thus mutexes provide a mechanism by which the programmer can control the serialization of multiple threads, ensuring that steps occur in a desired order and that the state corresponding to the mutex is maintained in a consistent manner" (col. 10, lines 11-22). This teaches the mutex lock is the guarantee that the thread holding the job will be executed and no data is lost. Hapner does not explicitly teaches providing a software mechanism that guarantees, as least to the primary level of guarantee, that the tasks will be executed without loss of data and without occurrence of any actual conflict with respect to the region of the database. Further more, McClaughry discloses providing a software mechanism that guarantees, as least to the primary level of guarantee, that the tasks will be executed without loss of data and without occurrence of any actual conflict with respect to the region of the database (There are also three types of write locks according to exemplary embodiments of the present invention. For example, in order to make changes to a file system object (e.g., update contents of a file), the application must acquire a Write Contents (WC) lock for that object. This can be accomplished by evaluating the lock record (if any exists) associated with the object that is to be changed as will be described below with respect to FIG. 5. If a lock record exists for this object with a WC flag set to indicate an existing WC lock (i.e., another thread is currently writing this object), then this lock request is denied and the user can be notified. Otherwise, if no lock record currently exists for that object, or if one exists without a WC lock indication, then the WC field is set equal to a value indicating that a WC lock has been acquired, e.g., one. Unlike the RC lock, however, the WC field in the lock record may be a boolean field since only one write operation is permitted on an object at a time. When a WC lock is acquired each of the ancestors of that object acquires a Hierarchical Write Contents (HWC) lock which indicates that a write operation is taking place on an object lower in the hierarchy. The field associated with HWC locks in a lock record is also formatted for counting since it is permissible for a write operation in one sub-branch to co-exist with a write operation in another sub_branch of the hierarchy. For example, if a WC lock is acquired for both files F and G in FIG. 2, folder C would have two HWC locks indicated in its lock record. Analogously to read locks, the third type of write lock is the Write Children (WK) lock. In conjunction wfth the RK lock, the V/K lock permits simultaneous writing of an object's contents and reading or writing of that object's children as will be described below with respect to FIG. 5.) (col. 6, lines 35-52). This read lock will resolve the conflicts and guarantee the write task to be executed without loss of data. Therefore, it would have been obvious to one ordinary skill in the art at the time of the invention was made to modify Hapner system to include write lock to resolve the conflict and to guarantee the write task to be completed without losing the data as taught by Hapner in order guarantee the write task to be completed without any interruption.

The applicant disagrees. McClaughry does not describe and would not have made obvious "providing a software mechanism that guarantees, as least to the primary level of guarantee, that the tasks will be executed."

On the contrary, McClaughry expressly forbids, rather than guarantees, the execution of two user requests that are concurrently incompatible, but could be sequentially compatible. (Col. 2, lines 62 – 65). One advantage of the combination recited in claim 68 is that, for example, multiple users can seek to commit transactions to a database, e.g., customer merchandise orders or customer address updates, with the confidence that the requests are guaranteed to be executed at a primary level of guarantee.

McClaughry, by contrast, is concerned with user requests in a single-user system. (Col. 3, lines 48 – 52). No combination of Hapner and McClaughry would have disclosed or suggested a system capable of "providing a software mechanism that guarantees, as least to the primary level of guarantee, that the tasks will be executed, and will be executed without loss of data and without the occurrence of any actual conflict with respect to the region of the database."

All of the dependent claims are patentable for at least similar reasons as those for the claims on which they depend are patentable.

Canceled claims, if any, have been canceled without prejudice or disclaimer.

Any circumstance in which the applicant has (a) addressed certain comments of the examiner does not mean that the applicant concedes other comments of the examiner, (b) made

arguments for the patentability of some claims does not mean that there are not other good reasons for patentability of those claims and other claims, or (c) amended or canceled a claim does not mean that the applicant concedes any of the examiner's positions with respect to that claim or other claims.

Applicant has corrected claim 57 so that the language is consistent with claim 57 as originally filed and shows amendments as filed in a reply dated June 27, 2007.

Please apply any other charges or credits to deposit account 06-1050, reference 11811-008002.

Respectfully submitted,

Date: 10/12/7

David L. Feigenbaum
Reg. No. 30,378

Fish & Richardson P.C.
225 Franklin Street
Boston, MA 02110
Telephone: (617) 542-5070
Facsimile: (617) 542-8906

21743737.doc